Multi-class Boosting for Imbalanced Data

Antonio Fernández-Baldera[1], José M. Buenaposada[2], and Luis Baumela[1]

[1] Dept. de Inteligencia Artificial, Universidad Politécnica de Madrid, Madrid, Spain
antonio.fbaldera@upm.es, lbaumela@fi.upm.es
[2] ETSII, Universidad Rey Juan Carlos, Madrid, Spain
josemiguel.buenaposada@urjc.es

Abstract. We consider the problem of multi-class classification with imbalanced data-sets. To this end, we introduce a cost-sensitive multi-class Boosting algorithm (*BAdaCost*) based on a generalization of the Boosting margin, termed *multi-class cost-sensitive margin*. To address the class imbalance we introduce a cost matrix that weighs more hevily the costs of confused classes and a procedure to estimate these costs from the confusion matrix of a standard 0|1-loss classifier. Finally, we evaluate the performance of the approach with synthetic and real data-sets and compare our results with the AdaC2.M1 algorithm.

1 Introduction

Imbalanced classification problems are characterized for having large differences in the number of samples in each class. This frequently occurs in complex data-sets, such as those involving class overlap, small sample size, or within-class imbalance. In this situation, standard classifiers perform poorly since they minimize the number of misclassified training samples disregarding minority classes [1]. Solutions to the class imbalance problem may be coarsely organized into data-based, that re-sample the data space to balance the classes, and algorithm-based approaches, that introduce new algorithms that bias the learning towards the minority class [1]. Boosting methods have been extensively used to address the problem of classification with imbalanced data-sets [1, 2] and cost-sensitive classification for two-class problems [3–5]. However, with the exception of AdaC2.M1 [2], no previous work has addressed the problem of multi-class Boosting in presence of imbalanced data.

In our proposal we merge both multi-class and cost-sensitive perspectives into a new Boosting algorithm, BAdaCost, that stands for *Boosting Adapted for Cost-matrix*. We introduce the concept of *Multi-class Cost-sensitive Margin*, which serves as link between multi-class margins and the values of the cost matrix, both of them considered as argument of a loss function. We also present a procedure to estimate this matrix from the confusion matrix of a standard 0|1-loss classifier. We justify BAdaCost's good properties in a set of experiments.

2 Background

In this section we briefly review some Boosting results related to our proposal. We start by introducing AdaBoost [6]. Given N training data instances $\{(\mathbf{x}_i, l_i)\}$, where $\mathbf{x}_i \in X$ encodes the object to be classified and $l_i \in L = \{+1, -1\}$ is the class label, the goal of AdaBoost is learning a strong classifier $\text{sign}(\mathtt{H}(\mathbf{x})) = \text{sign}(\sum_{m=1}^{M} \boldsymbol{\beta}_m G_m(\mathbf{x}))$ based on a linear combination of weak classifiers, $G_m : X \to L$. At each round m, a *direction* for classification, $G_m(\mathbf{x}) = \pm 1$, and a *step size*, $\boldsymbol{\beta}_m$ are added to an additive model whose goal is to minimize the empirical risk of the *Exponential Loss Function* [7], $\mathcal{L}(l, G_m(\mathbf{x})) = \exp(-l\, G_m(\mathbf{x}))$, defined over $z = l\, G_m(\mathbf{x})$, usually known as the *margin* [8].

2.1 Multi-class Boosting with Vectorial Encoding

A successful way to generalize the symmetry of class-label representation in the binary case to the multi-class case is using a set of vector-valued codes that represent the correspondence between the multi-class label set $L = \{1, \ldots, K\}$ and a collection of vectors $Y = \{\mathbf{y}_1, \ldots, \mathbf{y}_K\}$, where $\mathbf{y}_k$ has a value 1 in the kth coordinate and $\frac{-1}{K-1}$ elsewhere. It is immediate to see the equivalence between classifiers G defined over L and classifiers $\mathbf{g}$ defined over Y, $G(\mathbf{x}) = l \in L \;\Leftrightarrow\; \mathbf{g}(\mathbf{x}) = \mathbf{y}_l \in Y$. Zou, Zhu and Hastie [8] used this codification to generalize the concept of binary margin to the multi-class case using a related vectorial codification in which a K-vector $\mathbf{y} = (y_1, , \ldots, y_K)^\top$ is said to be a *margin vector* if it satisfies the *sum-to-zero* condition, $\sum_{i=1}^{K} y_i = \mathbf{0}$. The SAMME algorithm generalizes the binary AdaBoost to the multi-class case [9]. It uses the above codification and an exponential loss whose risk is minimized using a stage-wise additive gradient descent approach. In this loss function the binary margin, $z = lG(\mathbf{x})$, is replaced by the multi-class vectorial margin defined with a scalar product, $z = \mathbf{y}^\top \mathbf{g}(\mathbf{x})$, producing the *Multi-class Exponential Loss Function* (MELF), $\mathcal{L}(\mathbf{y}, \mathbf{g}(\mathbf{x})) = \exp\left(-\frac{\mathbf{y}^\top \mathbf{g}(\mathbf{x})}{K}\right)$.

In this paper we generalize the class-label representation here described so that our Boosting algorithm can model the asymmetries arising when training on an unbalanced data set.

2.2 Cost-Sensitive Binary Boosting

Classifiers that weigh certain types of errors more heavily than others are called cost-sensitive. They are used for example in medical diagnosis and object detection problems. Optimal cost-sensitive Boosting in two-class problems has been already studied in the literature [3–5]. The solution in [3] is based on minimizing the *Cost-sensitive Binary Exponential Loss Function* (CBELF) $\mathcal{L}(l, f(\mathbf{x})) = I(l = 1) \exp(-lC_1 f(\mathbf{x})) + I(l = -1) \exp(-lC_2 f(\mathbf{x}))$, where $I(\cdot)$ is the indicator function and C_j are the costs of the two possible errors. Classification with imbalanced data-sets is a typical cost-sensitive problem. To counter balance the bias in the data we want the classifier's loss function to under-weigh errors from

the majority class. In this paper we generalize the Cost-sensitive AdaBoost [3] to the multiple-class case and use it to solve imbalanced problems.

3 Multi-class Cost-Sensitive Margin

In this section we introduce the multi-class cost sensitive margin, based on which we derive the BAdaCost algorithm. Let us suppose the misclassification costs for our multi-class problem are encoded using a $K \times K$-matrix $\mathbf{C}$, where each value $C(i,j)$ represents the cost of misclassifying an instance with real label i as j. We can assume without loss of generality [10] that $C(i,i) = 0, \forall i \in L$, i.e. the cost of correct classifications is null. We introduce an essential change in the MELF to handle this kind of problems. Firstly, let $\mathbf{C}^*$ be a $K \times K$-matrix defined in the following way

$$C^*(i,j) = \begin{cases} C(i,j) & \text{if } i \neq j \\ -\sum_{h=1}^{K} C(j,h) & \text{if } i = j \end{cases}, \ \forall i,j \in L. \tag{1}$$

For our cost-sensitive classification problem each value $C^*(j,j)$ will represent a "negative cost" associated to a correct classification, i.e. a "reward".

The jth row in $\mathbf{C}^*$, denoted as $\mathbf{C}(j,-)$, is a margin vector that encodes the cost structure associated to the jth label. By using it we can define the *multi-class cost-sensitive margin value* for an instance $(\mathbf{x}, l)$ with respect to the multi-class vectorial classifier $\mathbf{g}(\cdot)$ as $z_C := \mathbf{C}^*(l,-) \cdot \mathbf{g}(\mathbf{x})$. It is easy to verify that if $\mathbf{g}(\mathbf{x}) = \mathbf{y}_i \in Y$, for a certain $i \in L$, then $\mathbf{C}^*(l,-) \cdot \mathbf{g}(\mathbf{x}) = \frac{K}{K-1}\mathbf{C}^*(l,i)$. Hence, multi-class cost-sensitive margins obtained from a discrete classifier $\mathbf{g} : \mathbf{x} \to Y$ can be computed using the "label valued" analogous of $\mathbf{g}$, $G : \mathbf{x} \to L$, $z_C = \mathbf{C}^*(l,-) \cdot \mathbf{g}(\mathbf{x}) = \frac{K}{K-1}\mathbf{C}^*(l, G(\mathbf{x}))$. We use this generalized margin as argument for the MELF in order to obtain the *Cost-sensitive Multi-Class Exponential Loss Function* (*CMELF*), $\mathcal{L}_C(l, \mathbf{g}(\mathbf{x})) := \exp(z_C) = \exp\left(\mathbf{C}^*(l,-) \cdot \mathbf{g}(\mathbf{x})\right)$, as the loss function for our problem. Although any other margin-based loss functions could have been used, we use the exponential loss to maintain the similarity with the original AdaBoost algorithm. The new margin, z_C, yields negative values when classifications are correct under the cost-sensitive point of view, and positive values far from costly (wrong) assignments. Moreover, the range of margin values of z_C is much broader than the $z = \pm 1$ values of AdaBoost.

The CMELF is a generalization of the MELF and CBELF. Let $\mathbf{C}_{0|1}$ be the cost matrix with zeros in the diagonal and ones elsewhere. This matrix encodes a multi-class problem free of costs. Further, it is well known that any matrix $\lambda\mathbf{C}_{0|1}$, with $\lambda > 0$, represents the same problem [10]. If we take into account that $\mathbf{y}^\top \mathbf{g}(\mathbf{x})$ has two possible values ($\frac{K}{K-1}$ when correct and $\frac{-K}{(K-1)^2}$ for errors) it is straightforward to prove that $\frac{1}{K(K-1)}\mathbf{C}_{0|1}$ will lead exactly to the same values of MELF when applied over the CMELF. In other words, the MELF is a particular case of the CMELF. On the other hand, it is also immediate to see that for a binary classification problem the values of $\mathbf{C}^*$ lead to the CBELF. Hence, it is a special case of CMELF as well.

Vectorial classifiers, $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_K(\mathbf{x}))^\top$, provides us with a *degree of confidence* for classifying sample $\mathbf{x}$ into every class. Hence, they use the max rule, $\arg\max_{k=1,\ldots,K} f_k(\mathbf{x})$, for label assignment [9, 11]. It is immediate to see that this criterion is equivalent to assigning the label that maximizes the multi-class margin, $\arg\max_{k=1,\ldots,K} \mathbf{y}_k^\top \mathbf{f}(\mathbf{x}) = \arg\min_{k=1,\ldots,K} -\mathbf{y}_k^\top \mathbf{f}(\mathbf{x})$. Since $-\mathbf{y}_k^\top \mathbf{f}(\mathbf{x})$ is proportional to $\mathbf{C}^*_{0|1}(k,-)^\top \mathbf{f}(\mathbf{x})$, we can extend the decision rule to the cost-sensitive field just by assigning $\arg\min_{k=1,\ldots,K} \mathbf{C}^*(k,-)\mathbf{f}(\mathbf{x})$.

4 BAdaCost: Boosting Adapted for Cost-Matrix

In this section we present the BAdaCost, a multi-class cost-sensitive Boosting algorithm. As we have defined the CMELF and given a training sample $\{(\mathbf{x}_i, l_i)\}$ we minimize the empirical expected loss, $\sum_{n=1}^N \mathcal{L}_C(l_n, \mathbf{f}(\mathbf{x}_n))$. The minimization is carried out by fitting an additive model, $\mathbf{f}(\mathbf{x}) = \sum_{m=1}^M \boldsymbol{\beta}_m \mathbf{g}_m(\mathbf{x})$. The weak learner selected at each iteration m will consists of an optimal step of size $\boldsymbol{\beta}_m$ along the direction $\mathbf{g}_m$ of the largest descent of the expected CMELF. In *Lemma 1* we show how to compute them.

Lemma 1. *Under the above assumptions, both $\boldsymbol{\beta}_m$ and $\mathbf{g}_m$ are given by minimizing:*

$$(\beta_m, \mathbf{g}_m(\mathbf{x})) = \arg\min_{\beta, \mathbf{g}(\cdot)} \sum_{j=1}^K \left(S_j \exp\left(\beta C^*(j,j)\right) + \sum_{k \neq j} E_{j,k} \exp\left(\beta C^*(j,k)\right) \right), \quad (2)$$

where the values of $S_j = \sum_{\{n: G(\mathbf{x}_n) = l_n = j\}} w_n$, $E_{j,k} = \sum_{\{n: l_n = j, G(\mathbf{x}_n) = k\}} w_n$ and $w_n = w_n exp\left(\boldsymbol{\beta}_m \mathbf{C}^*(l_n, -) \boldsymbol{g}_m(\mathbf{x}_n)\right)$. Given a known direction $\mathbf{g}$, the optimal step $\boldsymbol{\beta}(\mathbf{g})$ can be obtained as the solution to

$$\sum_{j=1}^K \sum_{k \neq j} E_{j,k} C^*(j,k) A(j,k)^\beta = -\sum_{j=1}^K S_j C^*(j,j) A(j,j)^\beta, \quad (3)$$

being $A(j,k) = \exp(C^*(j,k))$, $\forall i, j$. Finally, given a value of $\boldsymbol{\beta}$, the optimal descent direction $\mathbf{g}$, equivalently $G(\cdot)$, is

$$\arg\min_{G(\cdot)} \sum_{n=1}^N w_n \left(A(l_n, l_n)^\beta I\left[G(\mathbf{x}_n) = l_n\right] + \sum_{k \neq l_n} A(l_n, k)^\beta I\left[G(\mathbf{x}_n) = k\right] \right). \quad (4)$$

The BAdaCost pseudo-code is shown in Algorithm 1. At each iteration, we add a new multi-class weak learner $\mathbf{g}_m : X \to Y$ to the additive model weighted by $\boldsymbol{\beta}_m$, a measure of the confidence in the prediction of $\mathbf{g}_m$. The optimal weak learner that minimizes (Eq. 4) is a cost-sensitive multi-class classifier trained using the data weights, w_i, and a modified cost matrix, $\mathbf{C}_{wl}$, with $C_{wl}(i,j) = A(i,j)^\beta, \forall i, j$.

Algorithm 1. BAdaCost

Input: Cost matrix $\mathbf{C}$, N labeled training instances $(\mathbf{X}, \mathbf{Y})$ and number of iterations M
Output: The trained weak learners and weights (G_m, β_m), $m = 1, \ldots, M$

Initialize weight vector $\mathbf{w} \in \mathbb{R}^N$, with $w_i = 1/N$; $\forall i = 1, \ldots, N$.
$\mathbf{C}^* :=$ computeFullCostMatrix(C) { *Using equation (1)* }
for $m = 1, \ldots, M$ **do**
  $\beta := 1$; $c := \infty$; $\Delta c := \infty$.
  while $\Delta c \geq \gamma$ **do**
    $\mathbf{C}_{wl} :=$ computeWLCostMatrix($\mathbf{C}^*$, β).
    $G :=$ trainMulticlassCostSensitiveWL($\mathbf{X}$, $\mathbf{Y}$, $\mathbf{w}$, $\mathbf{C}_{wl}$).
    $\beta :=$ computeBeta($\mathbf{C}^*$, G, $\mathbf{w}$, $\mathbf{Y}$) { *Solving equation (3)* }
    $c_{new} :=$ computeCost($\mathbf{C}^*$, G, $\mathbf{w}$, $\mathbf{Y}$, β) { *Using β and G in equation (2)* }
    $\Delta c :=$ c - c_{new}; $c := c_{new}$.
  end while
  $G_m := G$; $\beta_m := \beta$.
  Translate G_m into $\mathbf{g}_m : X \rightarrow Y$.
  Update weights $w_i = w_i \exp\left(\beta_m \mathbf{C}^*(l_i, -)\mathbf{g}_m(\mathbf{x}_i)\right)$ for $i = 1, \ldots, N$, and re-normalize vector $\mathbf{w}$.
end for
Output Classifier: $\mathtt{H}(\mathbf{x}) = \arg\min_k \mathbf{C}^*(k, -)\mathbf{f}(\mathbf{x})$, where $\mathbf{f}(\mathbf{x}) = \sum_{m=1}^{M} \beta_m \mathbf{g}_m(\mathbf{x})$.

5 Experiments

In this section we experimentally evaluate BAdaCost's accuracy on imbalanced data-sets. In our experiments we use CART weak-learners and regularize our Boosting algorithm using shrinkage and re-sampling.

5.1 Cost Matrix Construction

A preliminary issue when using a cost-sensitive algorithm for solving an imbalance problem is establishing the cost matrix, $\mathbf{C}$. A straightforward solution would be to set the costs inversely proportional to the class imbalance ratios. However, this solution does not take into account the complexity of the classification problem. i.e. the amount of class overlap, within-class imbalance, etc. Here we introduce an alternative solution that considers the problem complexity. To this end we introduce a cost matrix that weighs more hevily the errors of poorly classified classes, hence the classifier will concentrate on the difficult minority classes.

Let $\mathbf{F}$ be the confusion matrix and $\mathbf{F}^*$ the matrix obtained when dividing each row i, $\mathbf{F}(i, -)$, by $\mathbf{F}(i, \cdot) = \sum_j F(i, j)$, i.e. the number of samples in class i. Then $F^*(i, j)$ is the proportion of data in class i classified as j. In a complex and imbalanced data-set, a $0|1$-loss classifier (e.g. BAdaCost with $0|1$-losses) will tend to over-fit the majority classes. So, off-diagonal elements in rows $\mathbf{F}^*(i, -)$ for majority (alt. minority) classes will have low (high) scores. Hence, the resulting matrix after setting $F^*(i, i) = 0, \forall i = 1 \ldots K$ is already a cost matrix. Finally, to improve numerical conditioning, we set $\mathbf{C} = \lambda \mathbf{F}^*$, for a small $\lambda > 0$.

5.2 Synthetic Datasets

The aim of this synthetic experiment is to visually analyze the performance of BAdaCost. We sample data from 2D Mixtures of Gaussian (MoG) probability distributions (see Fig. 1a) that represent a typical multi-class computer vision object detection problem. Two minority classes represent the target objects (colored areas in Fig. 1a) and a majority class that represents the background (black area in Fig. 1a). For training we sample 500 data for each class 1 (red) and 2 (green), and 5000 samples for class 3 (black). We also sample 1000 data from each class for testing. We run BAdaCost twice, firstly using a 0|1 cost matrix (see results in Fig. 1b) and in second place using a cost matrix built as described in Sect. 5.1 (see results in Fig. 1d).

In Fig. 1c we can see that, for 0|1 costs, training error evolves close to zero, whereas the testing error rate in a balanced data set levels-off above 0.4. This is an expected behavior, since this classifier optimizes the number of misclassified training samples, which come mostly from the background class, thus overfitting. Note here that, altough the classes are imbalanced, the error rate is a meaningful classification measure because the testing data set is balanced. When using a cost matrix, see Fig. 1e, we get a much better testing error rate of 0.2. The training error rate in this case is higher than that for the 0|1 cost matrix. This is also as expected, since the cost matrix has effectively moved the class boundary towards the majority class.

To visually appreciate the effect produced when training with an unbalanced data-set and the benefits of BAdaCost, in Fig. 1b and d we show respectively the result of classifying all points on a grid in the feature space of this problem with the 0|1 and the imbalanced cost matrix. We can see a much better reconstruction using the imbalanced cost matrix.

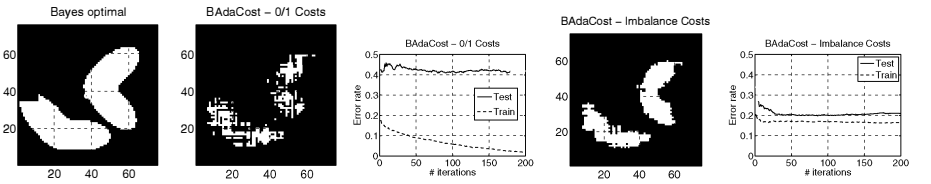

Fig. 1. Synthetic experiment 1. Simulated computer vision object detection problem with majority background class (black) (Color figure online).

5.3 Real Data-set: Synapse and Mitochondria Segmentation

In the last years we have seen advances in the automated acquisition of large series of images of brain tissue. The complexity of these images and the high number of neurons in a small section of the brain, makes the automated analysis

of these images the only practical solution. Mitochondria and synapses are two interesting cell structures that will be the object of detection. Unfortunately, the proportion of them w.r.t. the background is quite small, which makes the problem highly skewed. In our experiment we used an image stack obtained from the somatosensory cortex of a rat, with a resolution of 3.686 μm per pixel. The thickness of each layer is 20 μm [12]. From this data set we collected a training set composed of 10,000 background, 4000 mitochondria and 1000 synapse data and a testing set with 20,000 data per class.

In this section we use the BAdaCost algorithm to label pixels in these images as mitochondria, synapse and background, and compare the results with those achieved by the AdaC2.M1 algorithm. Following [12], we apply to each image in the stack a set of linear Gaussian filters at different scales to compute zero, first and second order derivatives. For each pixel we get a vector of responses $S = (s_{00}, s_{10}, s_{01}, s_{02}, s_{11}, s_{02})$ that are respectively obtained applying the filters $G_{\sigma *}, \sigma \cdot G_\sigma * \frac{\partial}{\partial x}, \sigma \cdot G_\sigma * \frac{\partial}{\partial y}, \sigma^2 \cdot G_\sigma * \frac{\partial^2}{\partial x^2}, \sigma^2 \cdot G_\sigma * \frac{\partial^2}{\partial xy}, \sigma^2 \cdot G_\sigma * \frac{\partial^2}{\partial y^2}$ where G_σ is a zero mean Gaussian with σ standard deviation. For a given σ the pixel feature vector is given by $f(\sigma) = (s_{00}, \sqrt{s_{10}^2 + s_{01}^2}, \lambda_1, \lambda_2)$ where λ_1 and λ_2 are the eigenvalues of the Hessian matrix of the pixel, that depend on s_{20}, s_{02} and s_{11}. The final 16 dimensional feature vector for each pixel is given by the concatenation of the $f(\sigma)$ vector at 4 scales (values of σ).

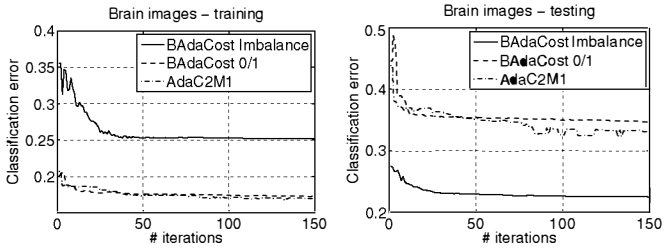

Fig. 2. Brain images experiment with a heavily imbalanced data-set.

In this experiment we compare BAdaCost (with 0|1 costs), AdaC2.M1 and BAdaCost (with the imbalanced cost matrix described in Sect. 5.1). In Fig. 2 we show the training and testing classification errors of the three algorithms. The behavior of BAdaCost in the real experiment is similar to those obtained with synthetic data. The 0|1-cost classifier has lower training error and higher testing error, whereas the the classifier with imbalanced cost matrix achieves the best generalization on the test set. The AdaC2.M1 classifier with imbalanced matrix achieve marginally better results than the 0|1-cost BAdaCost, but clearly worse than the imbalanced BAdaCost.

6 Conclusions

In this paper we have addressed the problem of multi-class classification with imbalanced data-sets. By extending the notion of multi-class margin to the cost-sensitive margin we introduced the *BAdaCost* algorithm and a procedure to estimate the cost matrix from the 0|1-loss confusion matrix. We have shown experimentally that BAdaCost performs as expected from a cost-sensitive algorithm and outperforms the AdaC2.M1 when dealing with imbalanced data.

Acknowledgments. This research was funded by the spanish *Ministerio de Economía y Competitividad*, project number `TIN2013-47630-C2-2-R`.

References

1. He, H., Garcia, E.A.: Learning from imbalanced data. IEEE Trans. Knowl. Data Eng. **21**(9), 1263–1284 (2009)
2. Sun, Y., Kamel, M.S., Wang, Y.: Boosting for learning multiple classes with imbalanced class distribution. In: Proceedings of the International Conference on Data Mining. ICDM '06, pp. 592–602 (2006)
3. Masnadi-Shirazi, Hamed, Vasconcelos, N.: Cost-sensitive boosting. Trans. Pattern Anal. Mach. Intell. **33**, 294–309 (2011)
4. Fan, W., Stolfo, S.J., Zhang, J., Chan, P.K.: Adacost: Misclassification cost-sensitive boosting. In: Proceedings of the 16th International Conference on Machine Learning, pp. 97–105 (1999)
5. Landesa-Vazquez, I., Alba-Castro, J.L.: Double-base asymmetric adaboost. Neurocomputing **118**, 101–114 (2013)
6. Freund, Y., Schapire, R.E.: A decision theoretic generalization of on-line learning and an application to boosting. J. Comput. Syst. Sci. **55**, 119–139 (1997)
7. Friedman, J., Hastie, T., Tibshirani, R.: Additive logistic regression: a statistical view of boosting. Ann. Stat. **28**(2), 337–407 (2000)
8. Zou, H., Zhu, J., Hastie, T.: New multicategory boosting algorithms based on multicategory fisher-consistent losses. Ann. Appl. Stat. **2**, 1290–1306 (2008)
9. Zhu, J., Zou, H., Rosset, S., Hastie, T.: Multi-class AdaBoost. Stat. Interface **2**, 349–360 (2009)
10. O'Brien, D.B., Gupta, M.R., Gray, R.M.: Cost-sensitive multi-class classification from probability estimates. In: Proceedings of the 25th International Conference on Machine Learning, pp. 712–719 (2008)
11. Fernandez-Baldera, A., Baumela, L.: Multi-class boosting with asymmetric weak-learners. Patt. Recogn. **47**(5), 2080–2090 (2014)
12. Cetina, K., Márquez-Neila, P., Baumela, L.: A comparative study of feature descriptors for mitochondria and synapse segmentation. In: Proceedings of the International Conference on Pattern Recognition (2014)